Please amend the first full paragraph starting on page 23 of the application as follows:

Referring now to FIGURE 20, there is illustrated a block diagram of the overall 3D engine. The core of the 3D engine is a rendering engine 2002, which is operable to receive video input data on a video input line 2004 which flows through a memory interface 2006 to the rendering engine 2002. The rendering engine 2002 is operable to receive the video input data in the form of various parameters, which parameters are to be converted into video data for output on a video output line 2008. This is a relatively conventional operation for rendering the pixels. The memory interface 2006 is also interfaced with two memories, a DRAM 2010 and an ~~SRAM 2012.~~ SDRAM 2012. The DRAM 2010 is utilized for storing the back buffer 1906, the Z-buffer 1902 and the anti-aliasing buffer 1904. The SDRAM 2012 is utilized for storing the front buffer 1908. Therefore, once the back buffer 1906 is created, it will be converted to a front buffer and transferred from the DRAM 2012 to the SDRAM 2012. During the filtering operation, the rendering engine 2002 is utilized in the filtering operation to operate on the SDRAM 2012 in conjunction with the information in the anti-aliasing buffer 1904 in the DRAM 2010. It is noted that, however, during the filtering operation, only the anti-aliasing mask is required for the filtering operation.

Please amend the first full paragraph starting on page 24 of the application as follows:

The triangle 2106 is disposed such that it only encloses three subpixels 2104 of the main pixel 2102. This particular triangle 2106 has associated therewith a depth value or a Z-value. During the rendering process, the pixel 2402 initially has a Z-value of infinity, due to the background being black. This is represented by the primary Z value Z_P, which is a 32-bit value. Essentially, this is represented by all the bits ~~being in~~ being a logical "1." After rendering, the depth value or the Z-value of the pixel 2102 will change, depending upon what polygon, a triangle in the present embodiment, has the center sampling point of the pixel 2102 enclosed therein. Therefore, the primary depth information, Z_P, of the pixel 2102 will ~~change to~~ change to the Z-value of the polygon. Additionally, during the rendering process, the number of subpixels that have the sampling point thereof disposed within the ~~triangle 2406~~ triangle 2106 will be determined so as to generate the anti-aliasing mask. This mask is indicated as

being the value "0000011000100000." A further measurement provided is a secondary Z-value, Z_S. This indicates the depth of the mask, which is determined from the triangle associated with that mask. For example, if the sampling point or ~~center point~~ TLC of the overall main pixel 2102 were enclosed within the triangle, the color of the pixel 2102 would be the same as the triangle and the depth of the mask would be that of the triangle. It is noted that this Z_S value is a 16-bit value, i.e., it is truncated from the 32-bit value associated with the triangle. This allows a 16-bit value to be stored in association with the 16-bit mask to provide a 32-bit overall anti-aliasing word. In another example, the pixel 2102 could be an edge pixel wherein the sampling point is outside of the triangle, such that the color of the pixel 2102 would be the color of the background but the depth of the mask would be that of the triangle 2106.

Please amend the last paragraph starting on page 24 and continuing to page 25 of the application as follows:

The purpose for the Z_S value is to ensure that there is only one mask value for each pixel, that being the mask value associated with the triangle that is nearest to the display. If, during rendering, it were determined that another triangle occupied all or a portion of the pixel ~~2402~~ 2102, the anti-aliasing mask for the pixel ~~2402~~ 2102 would be that associated with the one of the triangles which occupies at least one subpixel and that is nearest to the foreground, even if the center sampling point of the overall main pixel ~~2402~~ 2102 is outside of the nearest triangle.

Please amend the first full paragraph starting on page 25 and continuing to page 26 of the application as follows:

Referring now to FIGURE 22, there is illustrated an example of a deeper triangle that occupies a portion of the main pixel ~~2402~~ 2102. In this example, there is provided a triangle that is blue in color, the triangle 2406 being red in color and the background back in color, which larger triangle has an edge 2202 with the interior of the triangle indicated by arrows extending away therefrom. The larger triangle associated with the edge 2402 is disposed such that the pixel 2102 is an edge pixel, i.e., only a portion

of the subpixels 2104 disposed within the larger triangle, the blue triangle. However, the main Z-value for the large triangle is larger than that of the small triangle 2106, i.e., it is further away from the foreground. If the Z-value for the red triangle 2106 were 100 and the Z-value for the large blue triangle were 200, then it would indicate that the triangle 2106 were in the foreground. Therefore, when determining the main Z-value for the pixel 2102, the Z_P value, it is necessary to determine which polygon the sampling point, the "X" point, is disposed in. In this example of FIGURE 22, the sampling point is disposed within the blue triangle. Therefore, the Z-value for the pixel 2102 will be set to the color of the blue triangle and that leaves the decision as to what the anti-aliasing mask value should be. To determine this, the depth value for the blue triangle is compared to that of the Z_S value and the anti-aliasing buffer. If it is determined that the red triangle 2106 has a smaller Z value, associated with the Z_S value, then the mask will remain unchanged. However, if the blue triangle is nearer to the foreground, then the mask will be overwritten with the value correlating to the number of pixels having the sampling point thereof disposed within the blue triangle. This Z_S value is therefore utilized to resolve a coverage mask ordering problem for pixels that are partially covered. Although the pixel 2102 is illustrated as having the sampling point thereof disposed within the blue triangle, it could be that the blue triangle in fact partially covered the pixel 2102 but did not include the sampling point. Even in this situation, the Z_P of the pixel 2102 would be that of the background for some other color and would remain so. However, the mask value would be determined as that of the nearest triangle or polygon containing at least one subpixel (noting that a rule may be provided that more than one subpixel would be required for the triangle to have priority. Although the Z_S value is only a 16-bit value resulting is some loss of resolution in the decision process as to priority, this is a small tradeoff for the benefits provided as to mask ordering. Therefore, in the filtering or blending step, the color of the main pixel would be blended into neighboring pixels, depending upon the color values thereof and the mask values thereof. The values indicated for the mask indicate that the mask value has not changed, but that the Z_P-value has changed to the depth of the blue triangle, whereas the mask depth remains as that of the depth of the red triangle, i.e., the Z_S has not changed from one to the other, nor has the mask value changed, due to the fact that the triangle 2106 is nearer to the surface than the blue triangle.

Please amend the last paragraph starting on page 27 and continuing to page 28 of the application as follows:

Referring now to FIGURE 25, there is illustrated a flow chart depicting a filtering operation. The operation is initiated at a block 2502 and then proceeds to a function block 2504 wherein the rendering operation is performed utilizing the back buffer, the Z buffer and the anti-aliasing buffer. As lines are scanned, the back buffer is filled in with the color value for the designated pixel, i.e., the pixel being rendered, the Z value for the rendered pixel is determined, that being the Z value for the polygon ~~that~~ within which the sampling point of the rendered pixel resides and, if a determination is made that it is an edge pixel, then the mask value is determined. This is indicated by a function block 2506 wherein the anti-aliasing buffer, the back buffer and the Z buffer are filled. After completion of the rendering operation, then the back buffer must be transferred to the front buffer. In this operation, as indicated by function block 2508, the filtering operation performs a convolution operation wherein a 3x3 convolution kernel is utilized. This is comprised of a 3x3 array of pixels with a center pixel being the current pixel evaluated. By utilizing the anti-aliasing mask, and not the Z_S-value, the standard convolution process can be utilized to "blend" the current pixel with its neighboring pixels.

Please amend the first full paragraph starting on page 28 of the application as follows:

Referring now to FIGURE 26, there is illustrated a diagrammatic view of the filtering operation. There is illustrated the 3-D rendering engine core by reference numeral ~~2602~~ 2601. The 3-D core ~~2602~~ 2601 is operable to interface with a back buffer 2604 during the rendering operation to create the rasterized pixel. As described hereinabove, each pixel will have associated therewith a depth value and a color value, this color value comprised of the Red, Blue and Green color values. In addition to information in the back buffer 2604, there will also be provided an anti-aliasing (AA) buffer 2602 which is created during the rendering process. As described hereinabove, this AA buffer 2602 contains the anti-aliasing (AA) mask and also the secondary Z value, this secondary Z value representing the depth of the anti-aliasing mask for pixels that are partially covered by a polygon or triangle in the disclosed

embodiment, but are not considered inside the triangle, i.e., the sampling point is outside of the triangle. There is also provided a main Z-buffer 2606 which stores the Z values for the primary pixels.

Please amend the first paragraph on page 31 of the application as follows:

In the alternate condition wherein the center 2710 of the primary pixel 2702 lies outside of the triangle 2706, such that the line 2708 passes through a smaller number of subpixels 2704, this illustrated in FIGURE 28, the subpixels that lie within the triangle are set to a value of "0." The remaining subpixels 2704 are set to a value of "1." The value of "1" represents that these are the subpixels 2704 that are associated with space corresponding to the color of the primary pixel 2702. Initially, the sub-mask of subpixels 2704 is cleared to the value of "1" indicating that all of the subpixels are at the color of the primary pixel 2702, which is typically black, the primary background color. During rendering, select ~~one~~ ones of the subpixels 2704 are set to a value of "0" when it is determined that the sampling point thereof lies within a space of a different color. If the primary pixel lies mostly outside of the triangle (sampling point outside - center sampling point in this embodiment) and is the color of the background, the subpixels lying inside the triangle will have a "0" value and, if the primary pixel lies mostly inside of the triangle and is the color of the triangle, the subpixels lying outside of the triangle will have a "0" value. Initially, during the rendering process, the AA mask is created by setting the bit associated with the subpixels determined to lie within a triangle to a value of "1," regardless of how much of the primary pixel lies within the triangle. After creation of the AA mask, and before storage thereof in the AA mask, a determination is made as to whether the sampling point of the primary pixel is within the triangle or not. If outside of the triangle, then the bits n the AA mask are "flipped" and the color of the primary pixel set to the color of the space occupied by the sampling point of the primary pixel, the center thereof in this embodiment. As such, the value of "1" for a subpixel always indicates that it has a color associated with the color of the associated primary pixel. This will be utilized for the filtering process, as will be described hereinbelow.

Please amend the last paragraph starting on page 31 and continuing to page 32 of the application as follows:

Referring now to FIGURE 29, there is illustrated a simplified schematic of the color blending operation or filtering operation for one adjacent pixel to a center pixel 2902. The center pixel 2902 is the pixel that is being processed during the filtering operation. In this process, it can be seen that the mask value for the pixel 2902 is "1000110011101110." There is provided a right adjacent pixel 2904 and a left adjacent pixel 2905 that are utilized in the filtering operation with color blended into the center pixel 2902 from the adjacent pixels 2904 and 2905. As will be described hereinbelow, other adjacent pixels may be utilized, but only two are discussed in this example. The weight of the final color of the center pixel is set to the number of subpixels with a value of "1" multiplied by the color of pixel 2902 to provide a center result. When determining the weight of the adjacent pixels 2904 and 2905, the subpixels in primary pixel 2902 are evaluated that are in proximity thereto. In this example, the subpixel array is divided into two regions, a region 2906 comprised of the eight leftmost subpixels and a region 2907 associated with the eight rightmost subpixels. Region 2906 is used for blending the pixel 2905 and region 2907 is utilized for blending of the pixel 2904. In this example, the number of subpixels in the region 2906 having a "1" value are multiplied by the color of the pixel 2905 and a left result generated. Similarly, the number of subpixels in the region 2907 having a "1" value are multiplied by the color of the pixel 2904 and a right result generated. The right, left and center results are then added together and divided by 16 to provide a normalized result. As such, only a single value is required to determine how to blend color from adjacent pixels, due to the fact that the single value has not only percentage information associated therewith, but also directional information. This was also the case with the vector embodiment described hereinabove.